NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

DYNAMIC-PROGRAMMING
APPROACHES TO SINGLE- AND
MULTI-STAGE STOCHASTIC KNAPSACK
PROBLEMS FOR PORTFOLIO
OPTIMIZATION

by

Khoo, Wai Gea

March 1999

Advisor: R. Kevin Wood
Second Reader: David P. Morton

Approved for public release; Distribution is unlimited.

19990419 056

REPORT DOCUMENTATION PAGE		Form Approved OMB No. 0704-0188

1. AGENCY USE ONLY (*Leave blank*)	2. REPORT DATE March 1999	3. REPORT TYPE AND DATES COVERED Master's Thesis

4. TITLE AND SUBTITLE Dynamic-Programming Approaches to Single- and Multi-Stage Stochastic Knapsack Problems for Portfolio Optimization	5. FUNDING NUMBERS
6. AUTHORS Khoo, Wai Gea	

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey CA 93943-5000	8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution is unlimited.	12b. DISTRIBUTION CODE

13. ABSTRACT(*maximum 200 words*)

This thesis proposes new methods, based on dynamic programming, for solving certain single-stage and multi-stage integer stochastic knapsack problems. These problems model stochastic portfolio-optimization problems (SPOPs) which assume deterministic unit weight, and normally distributed unit return with known mean and variance for each item type. Given an initial wealth, the objective is to select a portfolio that maximizes the probability of achieving or exceeding a specified final return threshold; the multi-stage problem allows revisions of the portfolio at regular time intervals. An exact method is developed to solve a single-stage SPOP with independence of returns among item types. For a problem from the literature with 11 item types, this method obtains an optimal solution in a fraction of a second on a laptop computer. An approximation method, based on discretization of possible wealth values, is developed to solve a multi-stage SPOP with inter- and intra-stage independence of returns among item types. Running on a desktop computer, this approximation method solves a 3-stage problem with 6 item types in under 12 minutes. With finer discretization in a 3-stage problem with 8 item types, the solution time is about 46 minutes.

14. SUBJECT TERMS Dynamic Programming, Stochastic Programming, Knapsack Problem (Single-Stage, Multi-Stage), Portfolio Optimization	15. NUMBER OF PAGES 72
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT UL

DYNAMIC PROGRAMMING APPROACHES TO SINGLE- AND MULTI-STAGE STOCHASTIC KNAPSACK PROBLEMS FOR PORTFOLIO OPTIMIZATION

Khoo, Wai Gea
Ministry of Defence, Republic of Singapore
B.Eng. (Mechanical), National University of Singapore, 1994

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN OPERATIONS RESEARCH
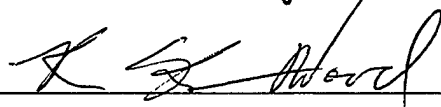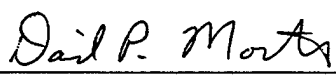
from the
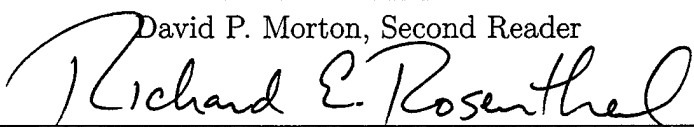
NAVAL POSTGRADUATE SCHOOL
March 1999

Author: _____
Khoo Wai Gea

Approved by: _____
R. Kevin Wood, Thesis Advisor

_____
David P. Morton, Second Reader

_____
Richard E. Rosenthal, Chairman
Department of Operations Research

iii

ABSTRACT

This thesis proposes new methods, based on dynamic programming, for solving certain single-stage and multi-stage integer stochastic knapsack problems. These problems model stochastic portfolio-optimization problems (SPOPs) which assume deterministic unit weight, and normally distributed unit return with known mean and variance for each item type. Given an initial wealth, the objective is to select a portfolio that maximizes the probability of achieving or exceeding a specified final return threshold; the multi-stage problem allows revisions of the portfolio at regular time intervals. An exact method is developed to solve a single-stage SPOP with independence of returns among item types. For a problem from the literature with 11 item types, this method obtains an optimal solution in a fraction of a second on a laptop computer. An approximation method, based on discretization of possible wealth values, is developed to solve a multi-stage SPOP with inter- and intra-stage independence of returns among item types. Running on a desktop computer, this approximation method solves a 3-stage problem with 6 item types in under 12 minutes. With finer discretization in a 3-stage problem with 8 item types, the solution time is about 46 minutes.

DISCLAIMER

The reader is cautioned that computer programs developed in this research may not have been exercised for all cases of interest. While every effort has been made, within the time available, to ensure that the programs are free of computational and logic errors, they cannot be considered validated. Any application of these programs without additional verification is at risk of the user.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF SYMBOLS

Symbols in bold font are the vector form of the variables or parameters represented by the corresponding symbols in regular font. For example, $\mathbf{r}$ is the vector form of r.

The index of an element in a vector is represented as a subscript. The variables, parameters and solution sets related to time t in a multi-stage problem are labeled with the subscript t; for elements of a vector, t is the index after the element index. For example, x_{kt} is the kth element of vector $\mathbf{x}$ at time t. Other subscripts and superscripts, which can be easily understood by the context in which they exist, are noted in the following list:

$_-$ $^-$ Lower and upper bounds, respectively, of the argument

$' \; '' \; \check{} \; \hat{}$ Intermediate or non-optimal representations of a variable or parameter

$*$ superscript indicating optimal solution of a problem

$+$ Subscript index of positive integer set; superscript index of an optimal objective value

$-$ Subscript index of negative integer set; superscript index of an optimal objective value

a Abitrary constant

c Desired minimum total return, or total return threshold

det Determinant of a matrix

E Mathematical expectation operator

f Function (optimal value function in a dynamic programming procedure)

h Probability density function

i Subscript index for total mean return

$\mathcal{I}$ Set of total mean in returns

j Subscript index for total variance in return

$\mathcal{J}$ Set of total variance in returns

k Subscript index of item type

K Number of item types; as a superscript, the dimension of a set

$\mathcal{K}$ Set of candidate portfolio items

l Index of an item among its type; quantity of an item type

L Maximum quantity affordable for an item type

$\mathcal{L}$ Set of affordable quantities of items

min   Subscript for a minimum value

n Superscript index for level of wealth realization

N Normal distribution; number of possible wealth realization levels

P Probability function

p Probability

r Unit return of an item

t Subscript index for time

T Length of planning horizon for a multi-stage problem; as a superscript, the transpose of a vector; as a subscript, the end of a planning horizon

U Utility value or function

$\mathcal{U}$ Set of values for total mean return

v Total or unit variance in return

$\mathcal{V}$ Set of values for total variance in return

w Weight or cost of an item

W Capacity of knapsack; available wealth for investment

x Quantity of items, which are of the same type, to include in the portfolio

$\mathcal{X}$ Solution set of a portfolio decision

y Decision (indicator) variable

Z The set of integers

$\mathcal{Z}$ Standard normal quantile

μ Total or unit mean return

Φ Standard normal cumulative distribution

ρ Objective function value

ζ Objective function value

LIST OF ABBREVIATIONS

DMSKP A dynamic-programming algorithm to a multi-stage stochastic knapsack problem

DP   Dynamic program

DSSKP A dynamic-programming algorithm to a single-stage stochastic knapsack problem

EMIP A function call to an explicit mixed-integer programming model

EMIPI An explicit mixed-integer programming model with integer mean, variance and portfolio constraints

EUSP A sub-problem maximizing an expected utility

IP   Integer program

KP   Knapsack problem

MAX A variance maximization problem

MAXVAR A variance maximization algorithm

MIN   A variance minimization problem

MIP   Mixed-integer program

MSKP0 A variant of a multi-stage stochastic knapsack problem

MSKP1 A variant of a multi-stage stochastic knapsack problem

NLP   Non-linear program

SDP   Stochastic dynamic program

SKP   Stochastic knapsack problem

SKP0 A variant of a single-stage stochastic knapsack problem

SKP1 A variant of a single-stage stochastic knapsack problem

SKP1a A variant of a single-stage stochastic knapsack problem

SPOP Stochastic portfolio-optimization problem

VMAX1 A variance maximization problem

EXECUTIVE SUMMARY

This thesis develops new methods for solving certain probabilistic versions of portfolio selection problems, in short, "stochastic portfolio-optimization problems." In these problems, each item type of the portfolio has deterministic unit cost, but probabilistic unit return value with known probability distribution. We assume a normal distribution in our study. Given an initial wealth, an investor would like to determine a portfolio with the best probability of achieving or exceeding a specified return threshold. The stochastic portfolio-optimization problems are closely related to "stochastic resource-allocation problems" which expend limited resources to acquire a system with maximized expected utility. Hence, the proposed solution techniques, with modifications, have wide applications in resource-allocation problems such as cargo loading, capital budgeting, project selection and weapons-mix problems.

The problems considered in this thesis assume that the returns for all item types are independent of each other. The first problem involves selection of a portfolio which cannot be altered until the end of an investment period when the portfolio is cashed; hence, it is called a "single-stage stochastic portfolio-optimization problem." We develop a method that provides an exact solution to this single-stage problem. This method examines each item type one by one, and considers different possible mixes of item-type quantities to yield specific mean and variance pairs for returns. Hence, the problem is decomposed into smaller and more manageable problems. At the examination of the last item type, an optimal portfolio, which has the highest probability of achieving or exceeding the return threshold, is selected. This stage-by-stage solution approach is a classical example of dynamic programming. For a problem from the literature with 11 item types, this method obtains an optimal solution in a fraction of a second on a laptop computer.

The problem just described involves a one-time decision. In practice, portfolios are revised at regular time intervals during the planning horizon. These stochastic

problems need decisions based on sequences of outcomes revealed over time; hence, these are multi-stage problems. We also consider, in this thesis, a multi-stage stochastic portfolio-optimization problem which assumes that the returns for all item types are independent of each other not only at a specific point in time, but also across time. An approximation method, based on an extension of the classical dynamic-programming technique, is developed for this multi-stage problem. Running on a desktop computer, this approximation method solves a 3-stage problem with 6 item types in under 12 minutes. A more precise approximate solution is obtained for a 3-stage problem with 8 item types in about 46 minutes. A possible improvement to our approximation method through the use of a sampling approach is also suggested at the end of the thesis.

In this thesis, we have shown the relevance and efficiency of dynamic-programming approaches to solving single- and multi-stage stochastic portfolio-optimization problems. Moreover, our multi-stage method, with modifications, has the potential to handle problems with certain dependencies among the item types. These dependencies add more realism to our problems, and extend them to possible future developmental work.

ACKNOWLEDGMENTS

I.   INTRODUCTION

This thesis proposes new methods, based on dynamic programming (DP), for solving some stochastic variants of the classical general integer knapsack problem. In particular, we consider single- and multi-stage stochastic portfolio-optimization problems (SPOPs) which assume deterministic unit weight, and normally distributed unit return with known mean and variance for each item type. Given an initial wealth, the objective is to select a portfolio that maximizes the probability of achieving or exceeding a final return threshold. We develop solutions for a single-stage (single-period) integer SPOP with independence of returns among the various item types in the portfolio, and a multi-stage (multi-period) integer SPOP with inter- and intra-period independence among item types. These problems, along with associated solution techniques, relate closely to single-stage stochastic knapsack problems, single- and multi-stage portfolio-optimization problems, multi-stage stochastic programs and stochastic dynamic programs. These problems are also related to stochastic resource-allocation problems. Hence, the proposed solution techniques, with modifications, have wide applications in problems such as cargo loading, capital budgeting, project selection and weapons-mix problems.

A.   BACKGROUND

This section provides some background on the single- and multi-stage stochastic knapsack problems (SKPs) which we solve in this thesis.

1.   Single-Stage Stochastic Knapsack Problems

In the classical knapsack problem, a hiker wishes to determine which of a set of items to carry on a backpacking trip. Each item has a weight and a "return value" to the hiker. Given a knapsack with limited weight capacity, the hiker wishes to determine the load to pack, so as to achieve the most valuable total return. This problem is also known as the flyaway kit problem (Taha 1992, pp. 358). It becomes

1

a general integer problem when the hiker can pack more than one unit of a particular item type.

In a stochastic variation of the knapsack problem, the returns from each item are random with known distributions. Random returns are common in the real world. For example, the returns of different financial instruments such as stocks and bonds, the revenue from a production plant or project, and the damage on a target resulting from a missile attack are all random in nature.

The integer SKP that we study here may be described as follows:

Indices

k item type, $k \in \mathcal{K} = \{1, 2, ..., K\}$

l index of an item among its own type, $l \in \{1, 2, ..., L_k\}$

Data

r_{kl} return for item l of type k; the marginal distribution of r_{kl} is assumed to be normal with mean μ_k and variance v_k, i.e., $r_{kl} \sim N(\mu_k, v_k) \ \forall \ k, l$;
$\mathbf{r} = (r_{11}, r_{12}, \ldots, r_{1L_1}, r_{21}, \ldots, r_{KL_K})^T$
(Different variants of SKP arise with different distributions for and dependency structure among the r_{kl}.)

w_k deterministic weight of each type-k item, $w_k \in Z_+ \ \forall \ k$

c desired minimum total return, i.e., the total return threshold

W capacity of the knapsack, $W \in Z_+$

Decision Variables

y_{kl} 1 if item l of type k is included in the knapsack; else 0
$\mathbf{y} = (y_{11}, y_{12}, \ldots, y_{1L_1}, y_{21}, \ldots, y_{KL_K})^T$

2

Formulation

GSKP(W)

$$\max_{\mathbf{y}} \quad P\left(\sum_{k=1}^{K}\sum_{l=1}^{L_k} r_{kl}y_{kl} \geq c\right)$$

$$\text{s.t.} \quad \sum_{k=1}^{K}\sum_{l=1}^{L_k} w_k y_{kl} \leq W$$

$$y_{kl} \in \{0,1\} \ \forall \ k \in \mathcal{K}, \ l = 1,2,...,L_k \ .$$

Here, $\sum_{l=1}^{L_k} y_{kl}$ is the number of items of type k to be included in the knapsack. The deterministic weight of each type k item is a positive integer w_k and W is the known, integer capacity of the knapsack. The returns $r_{k1}, ..., r_{kL_k}$ for a specific item type k are identically and normally distributed with mean μ_k and variance v_k. In some situations, this distributional form is reasonable, and furthermore, it leads to computationally tractable models.

The problem in GSKP(W) is to select an optimal $\mathbf{y}$ that maximizes the probability that the total return $\mathbf{r}^T\mathbf{y}$ meets or exceeds threshold c. Depending on the specific problem, there might be dependency in the unit return among items of the same type. In this thesis, we describe the extent of dependency among items of the same type with "complete independence," "partial dependence" and "complete dependence *within* item types." Similarly, for zero, partial or perfect dependency among the return from items of different types, we use the descriptions of "complete independence," "partial dependence" and "complete dependence *among* item types," respectively.

Carraway, Schmidt and Weatherford (1993) use "generalized dynamic programming" developed by Carraway, Morin and Moskowitz (1989) to solve GSKP(W) when there is complete independence among and within all item types. The more efficient techniques of Morton and Wood (1998) are applicable not only to this completely independent case, but also (with modifications) to situations where returns are identical within an item type, i.e., when there is complete dependence within item

types. In the latter case, the SKP corresponds to investment in multiple financial instruments such as stocks and bonds. The cost of each share of stock k is w_k dollars, the total wealth available for investment is W dollars, and the return from every share of stock k is r_k, which is a normal random variable. The objective is to invest limited assets so as to maximize the probability of achieving or exceeding a specified return. Hence, this model may be viewed as a simple, single-period SPOP; we develop a DP method for solving this problem in this thesis.

The portfolio-optimization version of the model GSKP(W) may be simplified to:

SKP0(W)

$$\max_{\mathbf{x}} \quad P\left(\sum_{k=1}^{K} r_k x_k \geq c\right)$$

$$\text{s.t.} \quad \sum_{k=1}^{K} w_k x_k \leq W$$

$$x_k \in Z_+ \ \forall \ k \in \mathcal{K}$$

where $r_k \equiv r_{k1} = r_{k2} = \cdots = r_{kL_k}$, w.p.1, and x_k is the number of type-k items to include in the portfolio.

The objective function expresses a variant of the "safety-first criterion" (Pyle and Turnovsky 1970). For this criterion, a disaster level of returns is first specified; the objective is to minimize the probability that the actual total return is worse (less) than or equal to the disaster level. The objective in SKP0(W) is essentially equivalent to this criterion.

2. Multi-Stage Stochastic Knapsack Problems

A single-stage problem involves a "one-time" decision. In practical planning problems such as production scheduling and power capacity expansion, it is not uncommon to find that multiple decisions are required at different phases of the planning horizon. These problems need decisions based on sequences of outcomes revealed over time (Birge and Louveaux 1997, pp. 128).

4

Consider a multi-stage problem expanded from the single-stage SKP0(W). The scenarios are standard to most multi-stage portfolio models: We begin with fixed capital, we invest in a set of financial instruments, we review and modify our investment portfolio after certain period of time has elapsed, and repeat this for a specified number of time periods. Our objective is to maximize the probability that we achieve or exceed a final target for total wealth (although other objectives are possible). We assume that no fee is charged for any transaction and no money is borrowed for investment. We are, therefore, only concerned about the uncertainty in the returns of the financial instruments. In addition, at the end of each time period, the portfolio rebalancing decision is only restricted by the available wealth accumulated from investments in the previous periods. The exact problem is defined in Chapter III.

The multi-stage SKP described above might be handled using the methods developed for the more general "multi-stage stochastic programming problems." Birge and Louveaux (1997, pp. 233-252) describe some of the exact methods that have been implemented with some success. Because exact methods are restricted to solving problems of moderate size and complexity, approximations (with deterministic or probabilistic bounds on accuracy) are often used to provide good solutions. Birge and Louveaux (1997, pp. 353-370) also discuss some of these techniques.

Much recent work on multi-stage stochastic programming involves developing models for financial planning problems faced by investment firms and corporations with large portfolios, e.g., insurance companies (Cariño *et al.* 1994). Such efforts include Klaassen's (1998) use of a state and time aggregation method in stochastic programming models for asset/liability management, and Hiller and Eckstein's (1993) use of massively parallel Benders decomposition in solving a stochastic portfolio model for fixed income asset/liability management. These models consider more complex economic factors than our problem, such as the uncertainty in interest rates and the need for liability management resulting from loans made. Hiller and Eckstein (1993)

5

adopt an efficient frontier approach (Sharpe 1970, pp. 52): this provides the portfolio manager with a set of portfolios that are efficient with respect to risk and return, rather than a portfolio that is optimal with respect to a single composite criterion. None of the above approaches fits well into our problem of determining an optimal initial portfolio decision based only on uncertain returns.

Stochastic dynamic programming (SDP) is a natural choice for our multi-stage SPOP. It is similar to deterministic dynamic programming except that decisions at a particular stage t depend on the realizations that have occured up to that point in time. A short discussion of SDP can be found in Kall and Wallace (1994, pp. 124-129).

B.   SCOPE

Morton and Wood (1998) state that their DP algorithm for the independent case may be extended to the dependent case, i.e., when there is complete dependence within item types. But those authors do not provide details. In Chapter II, we establish a variant of Morton and Wood's DP algorithm to handle the portfolio-optimization model SKP0(W) with complete dependence within item types, and complete independence among item types. In Chapter III, the model SKP0(W) is expanded to a sequential multi-stage decision problem. We then a new DP-like algorithm for solving this multi-stage portfolio-optimization model with complete inter- and intra-stage independence among the item types. Finally, we conclude this thesis and propose some future developmental work in Chapter IV.

II.   A SINGLE-STAGE STOCHASTIC KNAPSACK PROBLEM (SSKP)

In this chapter, we present a dynamic-programming solution to the single-stage portfolio-optimization model SKP0(W) with complete dependence within each item type as well as complete independence among item types. This solution is a variation of Morton and Wood's dynamic programming method (1998), which assumes complete independence within each item type. Hence, much of this chapter parallels that paper.

A.   MATHEMATICAL FORMULATION OF SSKP

The problem which we wish to solve is re-stated for clarity as follows:

Indices

k item type, $k \in \mathcal{K} = \{1, 2, ..., K\}$

Data

r_k return for each item of type k; the marginal distribution of r_k is assumed to be normal with mean μ_k and variance v_k, i.e., $r_k \sim N(\mu_k, v_k) \ \forall \ k$; $\mathbf{r} = (r_1, r_2, \ldots, r_K)^T$

w_k deterministic cost of each item of type k, $w_k \in Z_+ \ \forall \ k$

c desired minimum total return, i.e., the total return threshold

W initial total wealth, $W \in Z_+$

Decision Variables

x_k number of type-k items to include in the portfolio, $\mathbf{x} = (x_1, x_2, \ldots, x_K)^T$

Formulation

SKP0(W)

$$\max_{\mathbf{x}} \quad P\left(\sum_{k=1}^{K} r_k x_k \geq c\right)$$

$$\text{s.t.} \quad \sum_{k=1}^{K} w_k x_k \leq W$$

$$x_k \in Z_+ \ \forall \ k \in \mathcal{K}.$$

B. ASSUMPTIONS AND REFORMULATION

For the purpose of model formulation, we make the following assumptions:

1. The mean of all item types can be integerized through scaling and rounding with little loss of accuracy, when necessary. Therefore, we assume that $\mu_k \in Z_+ \ \forall \ k \in \mathcal{K}$.

2. The returns are independent among item types, i.e., $v_{ab} = 0 \ \forall \ a, b \in \mathcal{K}, \ a \neq b$, where v_{ab} denotes the covariance between the returns of item types a and b.

3. There is one and only one riskless item for the portfolio. A riskless item, classified to be type $k = 1$ throughout this thesis, has return $r_1 \sim N(\mu_1, 0)$ where $\mu_1 \geq w_1$.

4. The initial wealth $W \geq \min_{k \in \mathcal{K}} w_k$; this simply implies that the optimal knapsack will not be empty.

5. The threshold c is greater than the total return from a portfolio of riskless items, i.e., $c > \mu_1 \lfloor W/w_1 \rfloor$ where $\lfloor a \rfloor$ denotes the largest integer not exceeding a. Hence, the threshold c cannot be achieved with probability one.

6. There exists at least one feasible solution $\mathbf{x}$ with positive total variance. Thus, $P\left(\sum_{k=1}^{K} r_k x_k \geq c\right) > 0$.

The last three assumptions allow us to focus on portfolios with positive variance; portfolios with only riskless items, i.e., zero variance, are handled as special cases.

We now reformulate SKP0(W) using an equivalent deterministic objective function. For compact representation of the problem, the following vector notation is used:

8

unit returns, $\mathbf{r} = (r_1, r_2, \ldots, r_K)^T$,

portfolio, $\mathbf{x} = (x_1, x_2, \ldots, x_K)^T$,

unit cost, $\mathbf{w} = (w_1, w_2, \ldots, w_K)^T$,

unit mean returns, $\boldsymbol{\mu} = (\mu_1, \mu_2, \ldots, \mu_K)^T$, and

unit variance in returns, $\mathbf{v} = (v_1, v_2, \ldots, v_K)^T$.

Since $r_k \sim N(\mu_k, v_k)$, the total return of portfolio $\mathbf{x}$ is given by

$$\textstyle\sum_{k \in \mathcal{K}} r_k x_k = \mathbf{r}^T \mathbf{x} \sim N(\boldsymbol{\mu}^T \mathbf{x}, \mathbf{v}^T \mathbf{x}^2)$$

where $\mathbf{x}^2 = (x_1^2, x_2^2, \ldots, x_K^2)$. Therefore, we may convert the problem in SKP0(W) to a non-linear optimization problem. With the implicit constraint of positive total variance, i.e., $\mathbf{v}^T \mathbf{x}^2 > 0$, problem SKP0($W$) may be reformulated as

SKP1(W)

$$\rho^*(W) = \min_{\mathbf{x}} \quad (c - \boldsymbol{\mu}^T \mathbf{x})/\sqrt{\mathbf{v}^T \mathbf{x}^2}$$

$$\text{s.t.} \quad \mathbf{w}^T \mathbf{x} \leq W$$

$$\mathbf{x} \in Z_+^K.$$

The deterministic objective function is valid because

$$
\begin{aligned}
P\left(\mathbf{r}^T \mathbf{x} \geq c\right) &= P\left(\frac{\mathbf{r}^T \mathbf{x} - \boldsymbol{\mu}^T \mathbf{x}}{\sqrt{\mathbf{v}^T \mathbf{x}^2}} \geq \frac{c - \boldsymbol{\mu}^T \mathbf{x}}{\sqrt{\mathbf{v}^T \mathbf{x}^2}}\right) \\
&= P\left(N(0,1) \geq \frac{c - \boldsymbol{\mu}^T \mathbf{x}}{\sqrt{\mathbf{v}^T \mathbf{x}^2}}\right)
\end{aligned}
\tag{II.1}
$$

given $\mathbf{v}^T \mathbf{x}^2 > 0$, and therefore, the probability of achieving the return threshold c is maximized by minimizing the right-hand side of the inequality. This gives the objective function of SKP1(W).

C.   DYNAMIC PROGRAMMING FOR SSKP

In this section, we present a DP algorithm DSSKP to solve SKP1(W) or, equivalently, SKP0(W).

9

1. Concept

Suppose that $\underline{\mu}$ and $\bar{\mu}$ are, respectively, valid lower and upper bounds on $\boldsymbol{\mu}^T\mathbf{x}^*(W)$ where $\mathbf{x}^*(W)$ is an optimal solution to SKP1(W). Furthermore, assume that $\bar{\mu}$ is tight, i.e., $\bar{\mu} = \mu^*(W)$ where $\mu^*(W)$ is determined by solving a simple deterministic knapsack problem:

KP(W)

$$\mu^*(W) = \max_{\mathbf{x}} \quad \boldsymbol{\mu}^T\mathbf{x}$$

$$\text{s.t.} \quad \mathbf{w}^T\mathbf{x} \leq W$$

$$\mathbf{x} \in Z_+^K.$$

In this thesis, we set $\underline{\mu} = \min_{k \in \mathcal{K}} \mu_k$. For problems of larger scale, $\underline{\mu}$ can be set tighter. Now, define $\mathcal{U} = \{\underline{\mu}, \underline{\mu}+1, \ldots, \bar{\mu}\}$ and re-arrange SKP1(W) to obtain

SKP1a(W)

$$\rho^*(W) = \min_{\mu \in \mathcal{U}} \min_{\mathbf{x}} \quad (c - \mu)/\sqrt{\mathbf{v}^T\mathbf{x}^2}$$

$$\text{s.t.} \quad \boldsymbol{\mu}^T\mathbf{x} = \mu$$

$$\mathbf{w}^T\mathbf{x} \leq W$$

$$\mathbf{x} \in Z_+^K.$$

As noted by Morton and Wood (1998), for fixed $\mu < c$, $\rho^*(W)$ is minimized when $\mathbf{v}^T\mathbf{x}^2$ is maximized, and for fixed $\mu > c$, $\rho^*(W)$ is minimized when $\mathbf{v}^T\mathbf{x}^2$ is minimized. $\mu = c$ is a special case in which the objective value is zero. Consequently, there are three cases to consider for solving SKP1(W) based on the $\bar{\mu} = \mu^*(W)$ determined by KP(W).

a. Case 1, $\bar{\mu} < c$

In this case, for each $\mu \in \mathcal{U}$, we first solve

MAX(W, μ)

$$\zeta^+(\mu) = \max_{\mathbf{x}} \quad \mathbf{v}^T \mathbf{x}^2$$

$$\text{s.t.} \quad \boldsymbol{\mu}^T \mathbf{x} = \mu$$

$$\mathbf{w}^T \mathbf{x} \leq W$$

$$\mathbf{x} \in Z_+^K$$

for optimal solution $\mathbf{x}'(\mu)$. For any μ such that MAX(W, μ) is infeasible, we define $\zeta^+(\mu) = -\infty$. Then, the optimal objective value of SKP1a(W) is

$$\rho^*(W) = \min_{\mu \in \mathcal{U} | \zeta^+(\mu) > -\infty} (c - \mu)/\sqrt{\mathbf{v}^T \mathbf{x}'^2(\mu)}.$$

Hence, any solution in $\bigcup_{\mu \in \mathcal{U}} \{\mathbf{x}'(\mu)\}$ which satisfies $(c - \mu)/\sqrt{\mathbf{v}^T \mathbf{x}'^2(\mu)} = \rho^*(W)$ is an optimal solution to SKP1a(W) and therefore SKP1(W). It can be reasoned from (II.1) and the objective function of SKP1a(W) that, in this case,

$$0 < \rho^*(W) < \infty;$$

hence,

$$0 < P\left(\mathbf{r}^T \mathbf{x}^*(W) \geq c\right) < 0.5.$$

b. Case 2, $\bar{\mu} = c$

Since $\bar{\mu}$ is a tight upper bound, there exists a feasible solution $\mathbf{x}'(c)$ to SKP1a(W). Let $\mathbf{x}'(c)$ denote the solutions to KP(W) that achieved $\boldsymbol{\mu}^T \mathbf{x}'(c) = c$. This is an optimal solution to SKP1a(W) since it yields an objective value of zero which is higher than any of the objective values in Case 1 ($\mu < c$). Hence,

$$\rho^*(W) = 0,$$

and

$$P\left(\mathbf{r}^T \mathbf{x}^*(W) \geq c\right) = 0.5.$$

Note that the total variance of $\mathbf{x}^*(W)$ is irrelevant in this case. We ignore variance and take any optimal solution of KP(W).

c.   Case 3, $\bar{\mu} > c$

In this case, we first define the lower bound on μ as $\underline{\mu}' = \max\{\underline{\mu}, \lceil c \rceil\}$ (where $\lceil \cdot \rceil$ is the ceiling operator that returns the smallest integer not less than its argument). Then, we solve

MIN(W, μ)

$$\zeta^-(\mu) = \min_{\mathbf{x}} \quad \mathbf{v}^T \mathbf{x}^2$$

$$\text{s.t.} \quad \boldsymbol{\mu}^T \mathbf{x} \;=\; \mu$$

$$\mathbf{w}^T \mathbf{x} \;\leq\; W$$

$$\mathbf{x} \;\in\; Z_+^K$$

for $\mathbf{x}'(\mu)$, for each $\mu \in \mathcal{U}' = \{\underline{\mu}', \underline{\mu}' + 1, \ldots, \bar{\mu}\}$. Similar to MAX$(W, \mu)$, we define $\zeta^-(\mu) = \infty$ for any μ such that MIN(W, μ) is infeasible. Then, the optimal objective value of SKP1a(W) is

$$\rho^*(W) = \min_{\mu \in \mathcal{U}|\zeta^-(\mu) < \infty} (c - \mu)/\sqrt{\mathbf{v}^T \mathbf{x}'^2(\mu)}).$$

Hence, any solution in $\bigcup_{\mu \in \mathcal{U}'} \{\mathbf{x}'(\mu)\}$ which satisfies $(c - \mu)/\sqrt{\mathbf{v}^T \mathbf{x}'^2(\mu)}) = \rho^*(W)$ is an optimal solution to SKP1a(W) and therefore SKP1(W). In this case,

$$-\infty < \rho^*(W) < 0$$

$$\Leftrightarrow 0.5 < P\left(\mathbf{r}^T \mathbf{x}^*(W) \geq c\right) < 1.$$

2.   Algorithm Details

We use a three-phase algorithm to solve SKP1(W): Phase 1 solves KP(W) to determine which of Cases 1, 2 or 3 to consider for solving the problem. If Cases 1 or 3 are considered, Phase 2 solves a series of MAX(W, μ) or MIN(W, μ) problems. Phase 3 extracts and prints the optimal solutions determined.

a.   Phase 1

Phase 1 solves KP(W), a standard knapsack problem, using the following basic DP formulation (Dreyfus and Law 1977, pp. 108-110):

DP Recursion for solving KP(W)

Optimal Value Function

$\mu^*(w)$ = maximum total mean return attainable with available wealth w.

Recurrence Relation for $w = w_{\min}, w_{\min} + 1, \ldots, W$ where $w_{\min} = \min_{k \in \mathcal{K}} w_k$.

$$\mu^*(w) = \max_{k \in \mathcal{K}} \{\mu^*(w - w_k) + \mu_k\}.$$

Boundary Conditions

$$\mu^*(w) = \begin{cases} -\infty, & w < 0 \\ 0, & 0 \le w \le w_{\min} - 1. \end{cases}$$

Answer

$$\mu^*(W).$$

Now, if Phase 1 determines that $\mu^*(W) = c$, the algorithm will skip to Phase 3 where an optimal solution is extracted. Otherwise, Phase 2 is run to determine the maximum variance in returns by solving MAX(W, μ) for all $\mu \in \mathcal{U}$ if $\mu^*(W) < c$, or the minimum variance in returns by solving MIN(W, μ) for all $\mu \in \mathcal{U}'$ if $\mu^*(W) > c$.

b.   Phase 2

Let MAXE(w, μ) be the problem MAX(W, μ) with the constraint $\mathbf{w}^T\mathbf{x} \le W$ replaced by $\mathbf{w}^T\mathbf{x} = w$. In this phase, MAX($W, \mu$) for each $\mu \in \mathcal{U}$ are solved by first determining the solutions to MAXE(w, μ) for $w \in \{\underline{w}, \underline{w} + 1, \ldots, W\}$ where $\underline{w} = \min_{k \in \mathcal{K}} w_k$. After that, a solution that yields the lowest objective value of SKP1(W) is selected. In the same way, solutions of MIN(W, μ) for each $\mu \in \mathcal{U}'$ are determined by solving a series of problems MINE(w, μ) which is defined analogously to MAXE(w, μ).

The standard DP algorithm for the simple KP is extended to solved the two-constraint integer programs (IPs) MAXE(w, μ). In each stage of the algorithm,

only one item type is considered for knapsack loading. Hence, the index of the item types, k, is used as the stage number. The DP formulation is:

DP Recursion for solving MAXE(W, μ)

<u>Optimal Value Function</u>

$f_k(w', \mu') =$ maximum total variance in return from investing in item types $0, 1, \ldots, k$ given that wealth w' has been invested and a total return of μ' is expected. Item type 0 is a dummy item type for boundary definition.

<u>Recurrence Relation</u> for $k = 1, 2, \ldots, K$, $w' = w_{\min}, w_{\min} + 1, \ldots, w$, $\mu' = \mu_{\min}, \mu_{\min} + 1, \ldots, \mu$ where $w_{\min} = \min_{k \in \mathcal{K}} w_k$, $\mu_{\min} = \min_{k \in \mathcal{K}} \mu_k$

$$f_k(w', \mu') = \max_{x_k \in \{0, \ldots, \bar{x}_k\}} \{f_{k-1}(w' - w_k x_k, \mu' - \mu_k x_k) + v_k x_k^2\}$$

where $\bar{x}_k = \min\{\lfloor w'/w_k \rfloor, \lfloor \mu'/\mu_k \rfloor\}$.

<u>Boundary Conditions</u>

$$f_0(w', \mu') = -\infty, \quad w' \leq w_{\min} - 1, w' \neq 0, \mu' \leq \mu_{\min} - 1, \mu' \neq 0$$

$$f_k(0, 0) = 0, \quad k = 1, 2, \ldots, K.$$

<u>Answer</u>

$$f_K(w, \mu).$$

The above DP formulation may be modified to solve MINE(w, μ):

DP Recursion for solving MINE(W, μ)

<u>Optimal Value Function</u>

$f_k(w', \mu') =$ minimum total variance in return from investing in item types $0, 1, \ldots, k$ given that wealth w' has been invested and a total return of μ' is expected. Item type 0 is a dummy item type for boundary definition.

14

<u>Recurrence Relation</u> for $k = 1, 2, \ldots, K$, $w' = w_{\min}, w_{\min} + 1, \ldots, w$, $\mu' = \mu_{\min}, \mu_{\min} + 1, \ldots, \mu$ where $\underline{w} = \min_{k \in \mathcal{K}} w_k$, $\underline{\mu} = \min_{k \in \mathcal{K}} \mu_k$

$$f_k(w', \mu') = \min_{x_k \in \{0, \ldots, \bar{x}_k\}} \{f_{k-1}(w' - w_k x_k, \mu' - \mu_k x_k) + v_k x_k^2\}$$

where $\bar{x}_k = \min\{\lfloor w'/w_k \rfloor, \lfloor \mu'/\mu_k \rfloor\}$.

<u>Boundary Conditions</u>

$$f_0(w', \mu') = \infty, \quad w' \leq w_{\min} - 1, \ w' \neq 0, \ \mu' \leq \mu_{\min} - 1, \ \mu' \neq 0$$

$$f_k(0, 0) = 0, \quad k = 1, 2, \ldots, K.$$

<u>Answer</u>

$$f_K(w, \mu).$$

Phase 2 is divided into Phases 2a and 2b. Using either of the above recursions, Phase 2a determines the values $f_k(w, \mu)$ for $k \in \mathcal{K}$, $w \in \{w_{\min}, w_{\min} + 1, \ldots, W\}$, and $\mu \in \{\mu_{\min}, \mu_{\min} + 1, \ldots, \bar{\mu}\}$.

Subsequently, Phase 2b first redefines the lower bounds on μ and w: if $\mu^*(W) > c$, $\underline{\mu}' = \max\{\underline{\mu}, \lceil c \rceil\}$, and $\underline{w}' = \text{argmin}_{w = \{\underline{w}, \ldots, W\}}\{\mu^*(w) \geq c\}$; else, $\underline{\mu}' = \underline{\mu}$ and $\underline{w}' = \underline{w}$. Now, define SKPE$(w)$ as SKP1(W) with the constraint $\mathbf{w}^T\mathbf{x} \leq W$ replaced by $\mathbf{w}^T\mathbf{x} = w$. The optimal objective value $\rho(w)$ of SKPE(w) is determined by examining all finite values of $f_k(w, \mu)$ computed in Phase 2a for each $w \in \{\underline{w}', \underline{w}' + 1, \ldots, W\}$:

$$\rho(w) = \min_{\mu \in \mathcal{U}''(w), \, k \in \mathcal{K}} (c - \mu)/\sqrt{f_k(w, \mu)}.$$

where $\mathcal{U}''(w) = \{\underline{\mu}', \underline{\mu}'+1, \ldots, \mu^*(w) | \zeta^+(\mu) > -\infty\}$ for Case 1, or $\{\underline{\mu}', \underline{\mu}'+1, \ldots, \mu^*(w) | \zeta^-(\mu) < \infty\}$ for Case 3. The optimal objective value of SKP1(W) is then

$$\rho^*(w) = \min_{w \in \{\underline{w}', \ldots, W\}} \rho(w).$$

c. *Phase 3*

If Phase 1 determined that $\mu^*(W) = c$, this phase extracts the optimal solution to KP(W) as the optimal solution $\mathbf{x}^*(W)$ to SKP1(W). Otherwise, the optimal solution is extracted as the best solution to SKPE(w) over all $w \in \{w', w' + 1, \ldots, W\}$.

d. *Model Refinements*

To capitalize on the computer runtime to produce additional useful results, the algorithm DSSKP also extracts the optimal solutions $\mathbf{x}^*(w)$ to SKP1(w) for all $w \in \{w', w' + 1, \ldots, W\}$ in the final phase.

With insignificant extra effort, this additional information might be used to provide insight to a decision maker regarding the marginal loss of target-achievement probability due to loss in initial wealth.

To enhance computational efficiency, the recursive computations of $f_k(w, \mu)$ in Phase 2a and exhaustive search of $\rho(w)$ in Phase 2b are made only in the range with $\mu \leq \mu^*(w)$ for each w. Because $\mu^*(w)$ is the maximum feasible total mean return given initial weight w, it follows that $f_k(w, \mu)$ is invalid for values of μ outside this range; the exhaustive search in Phase 2b need not examine these f's.

3. Algorithm DSSKP

The DP algorithm for the SSKP (DSSKP) is presented as follows:

Algorithm DSSKP

Input: Data for SKP1(W) with K item types: integer vectors $\mathbf{w} > 0$, $\boldsymbol{\mu} > 0$, integer $W \geq \min_{k \in \mathcal{K}} w_k$, scalar c and real vector $\mathbf{v} \geq 0$.

Output (three possibilities):

1. If maximum feasible $\mu^*(W) = c$, optimal solution $\mathbf{x}^*(W)$ and solution value $\rho^*(W)$ to SKP(W);

2. if $\mu^*(W) < c$, optimal solutions $\mathbf{x}^*(w)$ and solution values $\rho^*(w)$ to SKP(w) for all $w \in \{w : \min_{k \in \mathcal{K}} w_k \leq w \leq W, w \in Z_+\}$;

3. if $\mu^*(W) > c$, optimal solutions $\mathbf{x}^*(w)$ and solution values $\rho^*(w)$ to SKP(w) for all $w \in \{w : \mu^*(w) \geq c, w \leq W, w \in Z_+\}$;

{

/* Phase 1 */

$\underline{w} \leftarrow \min_{k \in \mathcal{K}} w_k$;

$\mu^*(w) \leftarrow -\infty \ \forall \ w$ with $\underline{w} - \max_k w_k \leq w \leq W$;

$\mu^*(w) \leftarrow 0 \ \forall \ w$ with $0 \leq w \leq \underline{w} - 1$;

For $(w = \underline{w}$ to $W)$ {

     $k'(w) \leftarrow \text{argmax}_{k \in \mathcal{K}}\{\mu^*(w - w_k) + \mu_k\}$;

     $\mu^*(w) \leftarrow \mu^*(w - w_{k'(w)}) + \mu_{k'(w)}$;

}

If $(\mu^*(W) = c)$ go to Phase 3;

/* Phase 2a */

$\underline{\mu} \leftarrow \min_{k \in \mathcal{K}} \mu_k$; $\bar{\mu} \leftarrow \mu^*(W)$;

$w_{\min} - \min_k w_k$; $\mu_{\min} = \min_k \mu_k$;

If $(\mu^*(W) < c)$ {

     $f_k(w, \mu) \leftarrow -\infty \ \forall \ k, w, \mu$ with $0 \leq k \leq K$, $w_{\min} - \max_k w_k \leq w \leq W$, and $\mu_{\min} - \max_k \mu_k \leq \mu \leq \bar{\mu}$;

     $f_k(0, 0) \leftarrow 0 \ \forall \ k$ with $1 \leq k \leq K$;

     For $(k = 1$ to K and $w = w_{\min}$ to W and $\mu = \mu_{\min}$ to $\mu^*(w))$ {

         $\bar{x}_k \leftarrow \min\{\lfloor w/w_k \rfloor, \lfloor \mu/\mu_k \rfloor\}$;

         $x_k''(w, \mu) \leftarrow \text{argmax}_{x_k \in \{0, \ldots, \bar{x}_k\}}\{f_{k-1}(w - w_k x_k, \mu - \mu_k x_k) + v_k x_k^2\}$;

         $f_k(w, \mu) \leftarrow f_{k-1}(w - w_k x_k''(w, \mu), \mu - \mu_k x_k''(w, \mu)) + v_k x_k''^2(w, \mu)$;

     }

} else /* if $(\mu^*(W) > c)$ */ {

     $f_k(w, \mu) \leftarrow +\infty \ \forall \ k, w, \mu$ with $0 \leq k \leq K$, $w_{\min} - \max_k w_k \leq w \leq W$, and $\mu_{\min} - \max_k \mu_k \leq \mu \leq \bar{\mu}$;

     $f_k(0, 0) \leftarrow 0 \ \forall \ k$ with $1 \leq k \leq K$;

     For $(k = 1$ to K and $w = w_{\min}$ to W and $\mu = \mu_{\min}$ to $\mu^*(w))$ {

$$\bar{x}_k \leftarrow \min\{\lfloor w/w_k \rfloor, \lfloor \mu/\mu_k \rfloor\};$$

$$x_k''(w, \mu) \leftarrow \operatorname{argmin}_{x_k \in \{0, \dots, \bar{x}_k\}}\{f_{k-1}(w - w_k x_k, \mu - \mu_k x_k) + v_k x_k^2\};$$

$$f_k(w, \mu) \leftarrow f_{k-1}(w - w_k x_k''(w, \mu), \mu - \mu_k x_k''(w, \mu)) + v_k x_k''^2(w, \mu);$$

        }

}

/* Phase 2b */

If $(\mu^*(W) > c)$ {

    $\mu' \leftarrow \max\{\underline{\mu}, \lceil c \rceil\};$

    $\underline{w}' \leftarrow \operatorname{argmax}_{w=\underline{w}, \dots, W}\{\mu^*(W) \geq c\};$

} else {

    $\mu' \leftarrow \underline{\mu}; \quad \underline{w}' \leftarrow \underline{w};$

}

For $(w = \underline{w}'$ to $W)$ {

    $(\mu_t, k_t) \leftarrow \operatorname{argmin}_{\mu \in \mathcal{U}''(w), k \in \mathcal{K}}(c - \mu)/\sqrt{f_k(w, \mu)};$

    $\rho(w) \leftarrow (c - \mu_t)/\sqrt{f_{k_t}(w, \mu_t)};$

    $k'(w) \leftarrow k_t; \quad \mu'(w) \leftarrow \mu_t;$

    $\hat{w}(w) \leftarrow \operatorname{argmin}_{w_t \in \{\underline{w}', \dots, w\}}\rho(w_t);$

}

/* Phase 3 */

If $(\mu^*(W) = c)$ {

    $\mathbf{x} \leftarrow 0; \quad \hat{w} \leftarrow W;$

    do {

        $x_{k'(\hat{w})} \leftarrow x_{k'(\hat{w})} + 1;$

        $\hat{w} \leftarrow \hat{w} - w_{k'(\hat{w})};$

    } while $(\hat{w} \neq 0);$

    Print{ "Phase 1 shows $\mu^*($", $W,$ "$) = $", c};

    Print{ "Solution to SKP(W) for $W=$",$W,$ "is $\mathbf{x}^*(W) =$",$\mathbf{x}$};

    Print{ "with optimal objective value $\rho^*(W) =$",$\rho(W)$};

```
    } else {
        If (μ*(W) > c) Print{"Phase 1 shows μ*(",W, ") > ",c};
        else /* (μ*(W) < c) */ Print{"Phase 1 shows μ*(",W, ") < ",c};
        For (w = w' to W) {
            x ← 0;   ŵ ← ŵ(w);
            μ̂ ← μ'(ŵ);   k ← k'(ŵ);
            do {
                x_k ← x''_k(ŵ, μ̂);
                ŵ ← ŵ − w_k x_k;
                μ̂ ← μ̂ − μ_k x_k;
                k ← k − 1;
            } while(ŵ > 0);
            Print{"Solution to SKP(w) for w=",w,"is x*(w) =",x};
            Print{"with optimal objective value ρ*(w) =",ρ(w'(w))};
        }
    }
}
```

4. Computational Results

For testing the algorithm, we use the same data set as Morton and Wood (1998), i.e., the data from Steinberg and Parks (1979), with the addition of a riskless item type. This item type is item type 1 which has $w_1 = 1$, $\mu_1 = 1$ and $v_1 = 0$. Morton and Wood (1998) solves SKP0(W) with the assumption of complete independence within item types, whereas our method solves it with the assumption of complete dependence within item types. Despite this, our method is an extension of Morton and Wood's (1998) DP method (they denote it "DPSKP"). Hence, the relative computational time between their DPSKP and our DSSKP may provide a gauge for any extra complexity involved. The data are shown in Table I.

We use $W = 30$ and $c = 60$ (instead of $c = 30$ in the two works mentioned) for testing. In our case, $c = 30$ is illogical as investing in all riskless items will result in a total returns of 30 with certainty and meet the return threshold with probability

19

k	1	2	3	4	5	6	7	8	9	10	11
w_k	1	5	7	11	9	8	4	12	10	3	6
μ_k	1	7	12	14	13	12	5	16	11	4	7
v_k	0	15	20	15	10	8	20	8	15	20	25

Table I. Steinberg and Parks (1979) data with an additional riskless item.

one. Morton and Wood (1998) programmed their algorithm in Turbo Pascal and ran it on a Dell Latitude XPi laptop computer with a 133 MHz Pentium processor and 40 megabytes of RAM. Our algorithm is programmed in Java 1.1.2 and run in Microsoft's Windows 95 on a Dell Latitude LM laptop computer with a 166 MHz Pentium processor and 40 megabytes of RAM.

For our problem, it is determined in Phase 1 that $\mu^*(W) = 50$ (i.e., $\mu^*(W) < c$) and $w' = 1$. Hence, we obtain solutions for the range of $W = 1, 2, \ldots, 30$. The total solution time, which includes the data input and solution printing, is 0.44 seconds. We compare this to the total solution time of 0.026 seconds reported in Morton and Wood (1998) (for all values of W between 3 and 30). Our solution time is an order of magnitude greater than theirs although both are less than a second. This difference may partly be due to the use of different programming languages and different computers. Cases of $\mu^*(W) = c$ and $\mu^*(W) > c$ have also been tested by setting $W = 35$ and 50 respectively. The corresponding total solution times for these two cases are approximately 0.06 seconds and 0.94 seconds. Given the restricted time frame within which this thesis must be completed, we state these results without further comparative testing.

5. Comments

Similar to Morton and Wood's (1998) DPSKP algorithm, DSSKP is an exact method to solve an SPOP. Furthermore, it is simple to program. Based on the similar methodological extensions of basic DP algorithm as DPSKP, DSSKP can also be easily modified to accommodate bounded variables. Morton and Wood (1998) suggest that

this can be done by solving the bounded-variable version of SKPE(W) which is just a two-constraint, bounded-variable knapsack problem.

We have ignored cases in which $\mathbf{v}^T\mathbf{x}^2 = 0$ might be optimal. These special cases can be easily checked in Phase 2b of DSSKP. For these cases,

$$\frac{(c - \boldsymbol{\mu}^T\mathbf{x})}{\sqrt{\mathbf{v}^T\mathbf{x}^2}} = \begin{cases} -\infty & \text{if } \bar{\mu} \geq c, \\ \infty & \text{otherwise,} \end{cases}$$

$$\Leftrightarrow P(\mathbf{r}^T\mathbf{x} \geq c) = \begin{cases} 1 & \text{if } \bar{\mu} \geq c, \\ 0 & \text{otherwise.} \end{cases}$$

It may be of interest to the reader that it is possible to solve SKP1(W) by using the fact that for fixed $v = \mathbf{v}^T\mathbf{x}^2 > 0$, the objective function in SKP1a(W) for fixed v is minimized when $\boldsymbol{\mu}^T\mathbf{x}$ is maximized. In this case, the objective function of SKP1(W) can be solved using the re-arranged model

$$\rho^*(W) = \min_{v \in \mathcal{V}} \min_{\mathbf{X}} (c - \boldsymbol{\mu}^T\mathbf{x})/\sqrt{v},$$

where $\mathcal{V} = \{\underline{v}, \underline{v} + 1, \ldots, \bar{v}\}$ and $[\underline{v}, \bar{v}] \equiv$ the range of feasible total variances, $\underline{v} > 0$. But this approach is not advisable for solving SKP1(W) because, typically, $|\mathcal{V}| \gg |\mathcal{U}|$.

III. A MULTI-STAGE STOCHASTIC KNAPSACK PROBLEM (MSKP)

In this chapter, we present an approximate solution to a multi-stage portfolio-optimization model using a stochastic dynamic-programming (SDP) approach. This multi-stage problem assumes that portfolio revisions can be made at a finite number of points evenly spaced in time within a planning horizon of fixed length, i.e., an initial portfolio decision is made and rebalanced in stages. Within each stage of this multi-stage stochastic knapsack problem (MSKP), complete dependence of returns within item types is assumed. For returns among item types, there is complete independence within each stage and between stages, i.e., we assume complete intra- and inter-stage independence among item types.

A. MATHEMATICAL FORMULATION OF MSKP

This section presents a formulation of the MSKP using a DP-like recursion. In our MSKP, portfolio decisions are made at points in time, $t = 0, 1, \ldots, T-1$, and the final portfolio is evaluated at $t = T$. We define the interval of time between t and $t+1$ as period $t+1$. Hence, the portfolio selected at time t is fixed in period $t+1$ and revised at time $t+1$. Variables, parameters and solution sets related to time t are labeled with the subscript t; for elements of a vector indexed by t, the index t follows after the element index. For example, x_{kt} is the kth element of vector $\mathbf{x}_t$.

The formulation adopts the expected utility maxim (Markowitz 1959, pp. 205-242). Utility of a level of wealth indicates the satisfaction of an individual to that level of wealth. To maintain continuity with Chapter II, the utility of a realization of wealth at time T is defined to be zero unless it meets or exceeds the total return threshold c. The utility is one if wealth meets or exceeds the threshold, i.e., the portfolio's owner is fully satisfied with the wealth achieved. This wealth-utility relationship defines

the utility function at time T in our problem. A different utility function could be specified by the model user without changing our computational framework, although a few specialized techniques would be lost or require modification.

With a given available wealth at time $T - 1$, we intend to choose a portfolio that maximizes the probability of achieving a wealth of at least c at time T. The objective function at time $T - 1$ is, therefore, equivalent to determining a portfolio with the maximum expected utility at time T when the portfolio is cashed. Thus, we are using the expected utility maxim for our utility function in the last stage. In the model described below, this maxim is recursively applied to the portfolio revisions at all times t.

Indices

t possible transaction times within the planning horizon, $t \in \{0, 1, 2, \ldots, T\}$

Data

$\mathbf{w}_t$ unit cost vector of all item types at time t

W_0 initial wealth at time 0

c desired minimum total return, i.e., the total return threshold

$\mathbf{r}_t$ unit return vector of the items at time t for the portfolio selected at time $t - 1$; the distribution of $\mathbf{r}_t$ is multi-variate normal with mean vector $\boldsymbol{\mu}_t$ and variance vector $\mathbf{v}_t$; $\mathbf{r}_t = (r_{1t}, r_{2t}, \ldots, r_{Kt})^T$; $\boldsymbol{\mu}_t = (\mu_{1t}, \mu_{2t}, \ldots, \mu_{Kt})^T$; $\mathbf{v}_t = (v_{1t}, v_{2t}, \ldots, v_{Kt})^T$; hence, $r_{kt} \sim N(\mu_{kt}, v_{kt}) \; \forall \; k, t$

$h_t(W, \mathbf{r}_t^T \mathbf{x}_{t-1})$ probability density function for realizing wealth W with a one-period investment in portfolio $\mathbf{x}_{t-1}$ given a random return vector $\mathbf{r}_t$; we disallow negative portfolio returns so that $h_t(W, \mathbf{r}_t^T \mathbf{x}_{t-1}) \equiv 0$ for $W < 0$

Decision Variables

$\mathbf{x}_t$ portfolio decision at time t, $t = 0, 1, \ldots, T - 1$

$\mathcal{X}_t(W)$ set of feasible portfolio decisions at time t given an available wealth of W, $t = 0, 1, \ldots, T - 1$

$U_t(W)$ utility of realizing wealth W at time t, $t = 1, 2, \ldots, T$;

$$U_T(W) \equiv \begin{cases} 1 & \text{if } W \geq c, \\ 0 & \text{otherwise} \end{cases}$$

<u>Formulation</u>

MSKP0

$$U_0 = \max_{\mathbf{x}_0 \in \mathcal{X}_0(W_0)} E[U_1 | \mathbf{x}_0] \tag{III.1}$$

$$\text{s.t. } E[U_t | \mathbf{x}_{t-1}] = \int_W U_t(W) h_t(W, \mathbf{r}_t^T \mathbf{x}_{t-1}) dW \quad \text{for } t = 1, 2, \ldots, T \tag{III.2}$$

$$\text{where } U_t(W) = \max_{\mathbf{x}_t \in \mathcal{X}_t(W)} E[U_{t+1} | \mathbf{x}_t] \tag{III.3}$$
$$\text{for } t = 1, 2, \ldots, T-1 \text{ and } W \geq 0$$

$$\mathcal{X}_t(W) = \{\mathbf{x}_t \in Z_+^K | \mathbf{w}_t^T \mathbf{x}_t \leq W\} \tag{III.4}$$
$$\text{for } t = 1, 2, \ldots, T-1 \text{ and } W \geq 0$$

$$\mathcal{X}_0(W_0) = \{\mathbf{x}_0 \in Z_+^K | \mathbf{w}_0^T \mathbf{x}_0 \leq W_0\} . \tag{III.5}$$

The intention of the objective function (III.1) is to maximize the expected utility by selecting an optimal portfolio decision $\mathbf{x}_0^*$ for period 1 and a suite of optimal solution sets (optimal portfolio selection policy) $\mathcal{X}_t^*$ for $t = 1, 2, \ldots, T-1$. ($\mathbf{x}_0^*$ is commonly called the first-stage solution or decision.) Given the definition of the utility function at time T, the objective value U_0 is the probability of achieving or exceeding the return threshold c. Constraints (III.2) compute the expected utility, at each time t, of portfolio decision $\mathbf{x}_{t-1}$. Constraints (III.3) show that $U_t(W)$, the utility of wealth realization W at time t, is actually the maximum expected utility possible at time $t + 1$. This constraint and the objective function clearly illustrate

the recursive resolution of similar sub-problems to determine the final solution for the overall problem. The constraints (III.4) and (III.5) define feasible portfolios for wealth W at all time $t > 0$, and wealth W_0 at time $t = 0$ respectively.

B.   ASSUMPTIONS AND REFORMULATION

For the remainder of this chapter, we add the following assumptions to those of Chapter II:

1. The variance of all item types can be integerized through scaling and rounding with little loss of accuracy, when necessary. Therefore, we assume that $v_k \in Z_+ \; \forall \; k \in \mathcal{K}$.

2. The unit cost and return vectors at different times in the planning horizon are identical. Hence, $\mathbf{w} \equiv \mathbf{w}_0 = \mathbf{w}_1 = \cdots = \mathbf{w}_T$, and $\mathbf{r} \equiv \mathbf{r}_0 = \mathbf{r}_1 = \cdots = \mathbf{r}_T$. This is for notational simplification only.

3. Each riskless item has an unit cost of one, i.e., $w_1 = 1$.

4. The normal distribution for the total return is *left-censored at total return* $W = 0$, i.e.,

$$h_t(W, \mathbf{r}^T \mathbf{x}_{t-1}) = \begin{cases} 0 & \text{if } W < 0, \\ \int_{-\infty}^{0} \frac{1}{\sqrt{2\pi v}} e^{-(1/2v)(W-\mu)^2} \, dW & \text{if } W = 0, \\ \frac{1}{\sqrt{2\pi v}} e^{-(1/2v)(W-\mu)^2} & \text{otherwise,} \end{cases}$$

where $\mu = \boldsymbol{\mu}^T \mathbf{x}_{t-1}$ and $v = \mathbf{v}^T \mathbf{x}_{t-1}^2$.

5. The threshold c cannot be achieved or exceeded at the end of the planning horizon by making only riskless investments at all stages.

Now, we formulate an approximation of MSKP0. At each time t except $t = 0$, we discretize the realization of wealth $W \in [0, \infty)$ into N possible ranges:

$$
\begin{aligned}
[0, \infty] &= \left[0, \frac{c}{N}\right) \cup \left[\frac{c}{N}, \frac{2c}{N}\right) \cup \cdots \cup \left[\frac{(N-1)c}{N}, c\right) \cup [c, \infty) \qquad\qquad\text{(III.6)} \\
&\equiv \left[\underline{W}^1, \underline{W}^2\right) \cup \left[\underline{W}^2, \underline{W}^3\right) \cup \cdots \cup \left[\underline{W}^{N-1}, \underline{W}^N\right) \cup \left[\underline{W}^N, \underline{W}^{N+1}\right). \text{(III.7)}
\end{aligned}
$$

Any realized wealth W that falls in the range $[\underline{W}^n, \underline{W}^{n+1})$ will be mapped to a specified value W^n in that range. (In this thesis, variables, parameters and solution sets

related to the nth realization are labeled with the superscript n.) Of course, the choice of W^n will affect the solutions of the model. Solving the problem separately with the pessimistic choice $W^n = \underline{W}^n$ and the optimistic choice $W^n = \underline{W}^{n+1}$ will give, respectively, lower and upper bounds on the optimal solution value to MSKP0. (This works even though, strictly speaking, $W^n \equiv \underline{W}^{n+1} \notin [\underline{W}^n, \underline{W}^{n+1})$.) We take a "neutral approach" with $W^n = 0.5(\underline{W}^n + \underline{W}^{n+1})$ for $n = 1, \ldots, N - 1$ and $W^N = c$. However, in the computational tests, we demonstrate the possible bounds on the optimal solution value to MSKP0 by using the pessimistic and optimistic choices. The discretized formulation is

Indices

t possible transaction times within the planning horizon, $t \in \{0, 1, 2, \ldots, T\}$

n index for discretized level of wealth, $n = \{1, 2, \ldots, N\}$

Data

$\mathbf{w}$ unit cost vector for all item types

W_0 initial wealth at time 0

c desired minimum total return, i.e., the total return threshold

$[\underline{W}^n, \underline{W}^{n+1})$ nth range of wealth values as defined by equations (III.6) and (III.7)

$P^n(\mathbf{x})$ probability of realizing wealth $W \in [\underline{W}^n, \underline{W}^{n+1})$ from a one-period investment in portfolio $\mathbf{x}$; $P^n(\mathbf{x}) = P(\underline{W}^n \leq \mathbf{r}^T\mathbf{x} < \underline{W}^{n+1})$

W^n single representative value for wealth in range $[\underline{W}^n, \underline{W}^{n+1})$, $n = 1, 2, \ldots, N$

Decision Variables

$\mathbf{x}_t$ portfolio decision at time t, $t = 0, 1, \ldots, T - 1$

$\mathcal{X}_t^n$ feasible set of portfolio decisions given wealth W^n is available for investment at time t, $t = 1, 2, \ldots, T - 1$, $n = 1, 2, \ldots, N$

U_t^n utility of nth realization of wealth at time t, $t = 1, 2, \ldots, T-1$, $n = 1, 2, \ldots, N$; at time $t = T$, $U_T^n = \begin{cases} 1 & \text{if } n = N, \\ 0 & \text{otherwise} \end{cases}$

<u>Formulation</u>

MSKP1

$$U_0 = \max_{\mathbf{x}_0 \in \mathcal{X}_0} E[U_1 | \mathbf{x}_0] \qquad \text{(III.8)}$$

$$\text{s.t.} \quad E[U_t | \mathbf{x}_{t-1}] \quad = \quad \sum_{n=1}^{N} [P^n(\mathbf{x}_{t-1}) U_t^n] \quad \text{for } t = 1, 2, \ldots, T \qquad \text{(III.9)}$$

$$\text{where} \quad U_t^n \quad = \quad \max_{\mathbf{x}_t \in \mathcal{X}_t^n} E[U_{t+1} | \mathbf{x}_t] \qquad \text{(III.10)}$$
$$\text{for } t = 1, 2, \ldots, T-1 \text{ and } \forall\, n$$

$$\mathcal{X}_t^n \quad = \quad \{\mathbf{x}_t \in Z_+^K | \mathbf{w}^T \mathbf{x}_t \leq W^n\} \qquad \text{(III.11)}$$
$$\text{for } t = 1, 2, \ldots, T-1 \text{ and } \forall\, n$$

$$\mathcal{X}_0 \quad = \quad \{\mathbf{x}_0 \in Z_+^K | \mathbf{w}^T \mathbf{x}_0 \leq W_0\}\,. \qquad \text{(III.12)}$$

We develop a DP-like algorithm to solve the overall problem of MSKP1. In the algorithm, the sub-problems of MSKP1, as described by the objective function (III.8) and constraints (III.10), are formulated as mixed-integer programs (MIPs) and solved explicitly using a commercial MIP solver.

C.   DYNAMIC PROGRAMMING FOR THE OVERALL PROBLEM

In this section, we present a DP-like algorithm DMSKP to compute the recursion for MSKP1.

1.   Concept

The recursion in MSKP1 can be written in a DP formulation as:

28

DP Recursion for solving MSKP1

Optimal Value Function

U_t^n = maximum expected utility at the time $t + 1$ given that wealth W^n is available for investment.

Recurrence Relation for $t = 0, 1, \ldots, T - 1$, $n = 1, 2, \ldots, N$

For $t = 1, 2, \ldots, T - 1$,

$$U_t^n = \begin{cases} \max\limits_{\mathbf{x}_t \in \mathcal{X}_t^n} E[U_{t+1}|\mathbf{x}_t] & \text{for } n = 1, 2, \ldots, N - 1, \\ 1 & \text{for } n = N. \end{cases}$$

For $t = 0$,

$$U_0 = \max\limits_{\mathbf{x}_0 \in \mathcal{X}_0} E[U_1|\mathbf{x}_0].$$

Boundary Conditions

$$U_T^n = \begin{cases} 1 & \text{if } n = N, \\ 0 & \text{otherwise.} \end{cases}$$

Answer

U_0.

The resolution of each recursive relation can be handled by an explicit MIP formulation, which will be illustrated in the next section.

It is well known that the validity of DP is founded on the principle of optimality due to Bellman (any textbook on DP, such as Dreyfus and Law (1977), has an explanation of this principle). With respect to the DP formulation above, the principle of optimality simply states that if optimal U_0 is obtained, then each partial solution U_t^n obtained must be optimal for its respective state n and stage t, i.e., the optimal solution is composed of optimal partial solutions. This principle of optimality generally requires that the optimal value function be monotonic non-decreasing in sequential stages (the monotonicity condition). Hence, conventional DP, which we used in DSSKP, requires an additive optimal value function (or additive "accumulated

return function" as it is sometimes called) to ensure that monotonicity is satisfied. The validity of our proposed DP approach to MSKP holds under an extension of the monotonicity condition. Carraway *et al.* (1989) state this extended monotonicity condition and attribute it to Mitten (1964).

2. Algorithm DMSKP

DMSKP is a simple two-phase algorithm. The first phase defines boundary conditions, and then carries out a backward recursion from time $T - 1$ to 0, to determine the optimal objective values, U_t^n (or U_0), and solutions, x_t^{n*} (or x_0^*) for each realization of wealth W^n. Given t and n, if the deterministic total return from a portfolio with only riskless items meets or exceeds c, that portfolio is optimal with $U_t^n = 1$. Otherwise, U_t^n is obtained by solving a MIP. Formulation and solution of this MIP is represented by the function EMIP($\cdot$) in the algorithm. Phase 2 extracts and prints the results. The detailed sequence of DMSKP is:

Algorithm DMSKP

Input: Data for MSKP1 with K item types: integer vectors $\mu > 0$, $v \geq 0$, real vector $w > 0$, scalars $W_0 \geq \min_{k \in \mathcal{K}} w_k$, $c \geq \min_{k \in \mathcal{K}} w_k$, and $\{W^n, n = 1, 2, \ldots, N-1\}$.

Output: Optimal first-stage portfolio decision x_0^*, and solution sets $\mathcal{X}_t^* = \{x_t^{n*}, n = 1, 2 \ldots, N\}$ $\forall$ $t = 1, 2, \ldots, T-1$.

Function Called: EMIP(W^n, U_{t+1}, w, μ, v) determines an optimal portfolio decision x_t^* and its utility U_t^n at time t, given available wealth W^n, utility function values $U_{t+1} = (U_{t+1}^1, \ldots, U_{t+1}^N)$ and basic problem data.

{

    /* Phase 1 */

    $U_T^N \leftarrow 1$; $U_T^n \leftarrow 0$ $\forall$ $n = 1, 2, \ldots, N-1$;

    For ($t = T - 1$ downto 1) {

        For ($n = 1$ to $N - 1$) {

            $W' = 0.5(W^n + W^{n+1})$;

            If ($\mu_1 W' \geq c$)

$$U_t^n \leftarrow 1;$$
$$x_{1t}^{n*} \leftarrow W';$$
$$x_{kt}^{n*} \leftarrow 0 \ \forall \ k \neq 1;$$

else

$$(U_t^n, \mathbf{x}_t^{n*}) \leftarrow \text{EMIP}(W^n, \mathbf{U}_{t+1}, \mathbf{w}, \boldsymbol{\mu}, \mathbf{v});$$

}

}

$$U_t^N \leftarrow 1;$$
$$x_{1t}^{N*} \leftarrow W';$$
$$x_{kt}^{N*} \leftarrow 0 \ \forall \ k \neq 1;$$

}

$$(U_0, \mathbf{x}_0^*) \leftarrow \text{EMIP}(W_0, \mathbf{U}_1, \mathbf{w}, \boldsymbol{\mu}, \mathbf{v});$$

/* Phase 2 */

Print { "Solution to MSKP is:" };

Print { "$t = 0 : U = $ ", U_0, "$\mathbf{x}^* = $ ", $\mathbf{x}_0^*$};

For ($t = 1$ to $T - 1$ and $n = 1$ to N) {

Print { "$t = $", t, " $: U = $ ", U_t^n, "$\mathbf{x}^* = $", $\mathbf{x}_t^{n*}$};

}

}

D. EXPLICIT MIXED-INTEGER PROGRAMMING FOR THE SUB-PROBLEMS

This section describes an explicit MIP formulation to solve any sub-problem in MSKP1 and some proposed pre-processing steps to improve solution efficiency.

1. Model Formulation

Suppose a sub-problem requires an optimal portfolio decision at time t assuming that wealth W' has been achieved from the investment made at time $t - 1$. Extracted from MSKP1, the sub-problem is

31

EUSP

$$U_t^n = \max_{\mathbf{x}_t} E[U_{t+1}|\mathbf{x}_t]$$

$$\text{s.t. } \mathbf{w}^T\mathbf{x}_t \leq W'$$

$$\mathbf{x}_t \in Z_+^K.$$

Let the standard normal quantile related to $\underline{W}^n$ given a portfolio decision $\mathbf{x}$ be denoted as

$$\mathcal{Z}^n|\mathbf{x} = \frac{\underline{W}^n - \mu^T\mathbf{x}}{\sqrt{\mathbf{v}^T\mathbf{x}^2}},$$

where $\mathbf{v}^T\mathbf{x}^2 \neq 0$. Now, denoting the standard normal cumulative distribution as $\Phi(\cdot)$, we have, for $n = 1, 2, \ldots, N$,

$$P^n(\mathbf{x}) = \begin{cases} \Phi(\mathcal{Z}^2|\mathbf{x}) & \text{if } n = 1, \\ \Phi(\mathcal{Z}^{n+1}|\mathbf{x}) - \Phi(\mathcal{Z}^n|\mathbf{x}) & \text{if } n = 2, 3, \ldots, N-1, \\ 1 - \Phi(\mathcal{Z}^N|\mathbf{x}_t) & \text{otherwise,} \end{cases}$$

based on the assumption of left-censored normal distribution for the total return. For $\mathbf{v}^T\mathbf{x}_t^2 \neq 0$, the objective function of EUSP can, therefore, be re-written as

$$
\begin{aligned}
E[U_{t+1}|\mathbf{x}_t] &= \sum_{n=1}^{N} P^n(\mathbf{x}_t)U_{t+1}^n \quad \text{(from (III.9))} \\
&= \Phi(\mathcal{Z}^2|\mathbf{x}_t)U_{t+1}^1 + \sum_{n=2}^{N-1} \left[\Phi(\mathcal{Z}^{n+1}|\mathbf{x}_t) - \Phi(\mathcal{Z}^n|\mathbf{x}_t)\right]U_{t+1}^n \\
&\quad + \left[1 - \Phi(\mathcal{Z}^N|\mathbf{x}_t)\right]U_{t+1}^N \\
&= -\sum_{n=1}^{N-1}\left(U_{t+1}^{n+1} - U_{t+1}^n\right)\Phi(\mathcal{Z}^{n+1}|\mathbf{x}_t) + U_{t+1}^N \\
&= 1 - \sum_{n=1}^{N-1}\left(U_{t+1}^{n+1} - U_{t+1}^n\right)\Phi(\mathcal{Z}^{n+1}|\mathbf{x}_t) \quad \text{since } U_{t+1}^N = 1.
\end{aligned}
$$

If $\mathbf{v}^T \mathbf{x}_t^2 = 0$ and $\mathbf{x}_t \neq \mathbf{0}$,

$$E\left[U_{t+1} | \mathbf{x}_t\right] = U_{t+1}^n \text{ such that } \underline{W}^n \leq \boldsymbol{\mu}^T \mathbf{x}_t < \underline{W}^{n+1}.$$

Furthermore, we observe that the expected utility is a direct function of the mean and variance of the total portfolio return. Hence, we will also focus on finding the best mean-variance combination for the total portfolio return. We present our MIP formulation for EUSP:

Indices and Index Sets

t index for the time at which an optimal portfolio decision is required

n index for level of wealth, $n \in \{1, 2, \ldots, N\}$

k index for item type, $k \in \mathcal{K} = \{1, 2, \ldots, K\}$

i index for mean of total return, $i \in I$

j index for variance of total return, $j \in \mathcal{J}$

$\mathcal{J}_i$ index set of variance values corresponds to a total mean with index i (Note: $\mathcal{J} \equiv \bigcup\limits_{i \in \mathcal{I}} \mathcal{J}_i.$)

l index for quantity of an item type

$\mathcal{L}_k$ index set of possible affordable quantities for type-k item, $\mathcal{L}_k \in \{0, 1 \ldots, L_k\}$ where L_k is the maximum number of type-k items affordable given an available wealth at time t

n_i' index for the level of wealth in which a mean total return with index i falls, $n_i' \in \{1, 2, \ldots, N\}$ (see below for further description)

Data

$\check{\mu}_i$ the ith level of mean total return, $\check{\mu}_i = 0, 1, \ldots, \bar{\mu}$

$\check{v}_j$ the jth level of total variance in return, $\check{v}_j = 0, 1, \ldots, \bar{v}$

$\check{x}_l$ quantity of item to invest; $\check{x}_l = l$

μ_k unit mean return of item type k

v_k unit variance in return of item type k

w_k unit cost of item type k

W' available wealth at time t

$[\underline{W}^n, \underline{W}^{n+1})$ nth range of wealth values as defined by equations (III.6) and (III.7) (Note: For a portfolio with mean total return $\breve{\mu}_i$, n'_i denotes n such that $\underline{W}^n \leq \breve{\mu}_i < \underline{W}^{n+1}$}.)

U^n_{t+1} optimal utility of achieving wealth level n at time $t+1$

Decision Variables

y'_{ij} binary indicator variable:

$$y'_{ij} = \begin{cases} 1 & \text{if portfolio with } (\breve{\mu}_i, \breve{v}_j) \text{ mean-variance combination is selected,} \\ 0 & \text{otherwise} \end{cases}$$

y''_{kl} binary indicator variable:

$$y''_{kl} = \begin{cases} 1 & \text{if } \breve{x}_l \text{ units of type-}k \text{ item are invested in,} \\ 0 & \text{otherwise} \end{cases}$$

Formulation

EMIPI

$$U^n_t = \max_{\mathbf{y'}, \mathbf{y''}} \ 1 - \sum_{i \in \mathcal{I}} \left\{ \sum_{j \in \mathcal{J}_i | \breve{v}_j \neq 0} \sum_{n=1}^{N-1} \left[(U^{n+1}_{t+1} - U^n_{t+1}) \Phi \left(\frac{\underline{W}^{n+1} - \breve{\mu}_i}{\sqrt{\breve{v}_j}} \right) \right] y'_{ij} \right. \tag{III.13}$$

$$\left. + \sum_{j \in \mathcal{J}_i | \breve{v}_j = 0} U^{n'_i}_{t+1} y'_{ij} \right\}$$

$$\text{s.t.} \quad \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{J}_i} \breve{\mu}_i y'_{ij} - \sum_{k \in \mathcal{K}} \sum_{l \in \mathcal{L}_k} \mu_k \breve{x}_l y''_{kl} = 0 \tag{III.14}$$

$$\sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{J}_i} \breve{v}_j y'_{ij} - \sum_{k \in \mathcal{K}} \sum_{l \in \mathcal{L}_k} v_k \breve{x}_l^2 y''_{kl} = 0 \tag{III.15}$$

$$\sum_{k \in \mathcal{K}} \sum_{l \in \mathcal{L}_k} w_k \breve{x}_l y''_{kl} \leq W' \tag{III.16}$$

$$\sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{J}_i} y'_{ij} = 1 \tag{III.17}$$

$$\sum_{l \in \mathcal{L}_k} y''_{kl} = 1 \ \forall \ k \in \mathcal{K} \tag{III.18}$$

$$y'_{ij} \in \{0,1\} \ \forall \ i \in \mathcal{I}, \ j \in \mathcal{J}_i$$

$$y''_{kl} \in \{0,1\} \ \forall \ k \in \mathcal{K}, \ l \in \mathcal{L}_k .$$

34

Note that if minimum and/or maximum investment levels must be specified for any item type, this can be accommodated in EMIPI, and thus in MSKP1, by modifying the definition of the set $\mathcal{L}_k$. Furthermore, general constraints on the original x_k can be added to the model, albeit clumsily.

The objective function (III.13) maximizes expected utility in EUSP. Although the constant 1 in the objective function is unnecessary, it is kept to clarify the link between EMIPI and EUSP. Constraints (III.14) and (III.15) ensure that the portfolio decision y''_{kl} matches the total mean-variance combination selected by the indicator variable y'_{ij}. Constraint (III.16) limits investments by total available wealth. Constraint (III.17) specifies that only one total mean-variance combination can be chosen as the solution. In a similar manner, Constraint (III.18) restricts the model to select only a single quantity to invest for each item type.

2. Preprocessing

If we think of an i-j combination as a grid point, it is clear that the possible i-j combinations in EMIPI form a vast search matrix for a MIP solver to find an optimal y'_{ij}. This matrix expands pseudo-polynomially with the order of magnitudes of the feasible total mean and variance in return; and the solution time of EMIPI explodes exponentially with the problem size. Hence, reduction in the size of the search matrix is necessary to enable solution of even modest-sized problems. Let $\mathcal{IJ}$ be the set of candidate i-j combinations for search consideration in EMIPI. We suggest the following preprocessing steps to reduce the size of $\mathcal{IJ}$ and prepare data before solving EMIPI explicitly with a MIP solver:

a. Step 1

First, for fixed wealth W', we wish to determine the set of feasible total mean returns. A valid lower bound on feasible total return is defined by

$$\underline{\mu} = \min_{k \in \mathcal{K}} \left[\mu_k \left\lfloor \frac{W'}{w_k} \right\rfloor + \mu_1 \mathrm{mod}(W', w_k) \right]$$

where $\mathrm{mod}(a, b)$ denotes the remainder of a/b.

The upper bound $\bar{\mu}$ on the total mean return is defined by the optimal objective value of the knapsack problem KP(W'). Then, we define index set $\mathcal{I} = \{i \mid \text{For some } \check{\mu}_i \in [\underline{\mu}, \bar{\mu}], \exists \, \mathbf{x} \in Z_+^K \text{ such that } \boldsymbol{\mu}^T\mathbf{x} = \check{\mu}_i \text{ and } \mathbf{w}^T\mathbf{x} = W'\}$.

b. Step 2

Here, we determine for each $i \in \mathcal{I}$, a corresponding bounded range for total variance, $[\underline{v}_i, \bar{v}_i]$ for a feasible portfolio with $\boldsymbol{\mu}^T\mathbf{x} = \check{\mu}_i$. The corresponding index set is denoted by $\mathcal{J}_i$. A valid lower bound $\underline{v}_i$ could be found by solving a convex non-linear problem:

VMIN0

$$\underline{v}_i = \left\lceil \min_{\mathbf{x}} \quad \mathbf{v}^T\mathbf{x}^2 \right\rceil$$

$$\begin{aligned}
\text{s.t.} \quad \boldsymbol{\mu}^T\mathbf{x} &= \check{\mu}_i \\
\mathbf{w}^T\mathbf{x} &= W' \\
\mathbf{x} &\geq 0,
\end{aligned}$$

Or, we can add integer restrictions on the $\mathbf{x}$ and solving this model which is similar in philosophy to EMIPI:

VMIN1

$$\underline{v}_i = \min_{y''} \sum_{k \in \mathcal{K}} \sum_{l \in \mathcal{L}_k} v_k \check{x}_l^2 y_{kl}''$$

$$\begin{aligned}
\text{s.t.} \quad \sum_{k \in \mathcal{K}} \sum_{l \in \mathcal{L}_k} \mu_k \check{x}_l y_{kl}'' &= \check{\mu}_i \\
\sum_{k \in \mathcal{K}} \sum_{l \in \mathcal{L}_k} w_k \check{x}_l y_{kl}'' &= W' \\
\sum_{l \in \mathcal{L}_k} y_{kl}'' &= 1 \; \forall \, k \in \mathcal{K} \\
y_{kl}'' &\in \{0,1\} \; \forall \, k \in \mathcal{K}, \, l \in \mathcal{L}_k.
\end{aligned}$$

In practice, we use VMIN0 rather than VMIN1 because the tradeoff between computation speed and bound quality is won by the continuous model in our computational tests.

There is an exceptional situation when we solve VMIN1 to get tight lower bounds: Note that for fixed i such that $\breve{\mu}_i \geq c$, $\dfrac{\underline{W}^{n+1} - \breve{\mu}_i}{\sqrt{\breve{v}_j}} \leq 0$ for $n = 1, 2, \ldots, N-1$ and $\breve{v}_j \in [\underline{v}_i, \bar{v}_i]$. For such an index i, the objective function of EMIPI

$$1 - \left\{ \sum_{j \in \mathcal{J}_i | \breve{v}_j \neq 0} \sum_{n=1}^{N-1} \left[(U_{t+1}^{n+1} - U_{t+1}^n) \Phi \left(\frac{\underline{W}^{n+1} - \breve{\mu}_i}{\sqrt{\breve{v}_j}} \right) \right] y'_{ij} + \sum_{j \in \mathcal{J}_i | \breve{v}_j = 0} U_{t+1}^{n'} y'_{ij} \right\}$$

is maximized when $\breve{v}_j$ is minimized. Therefore, $[\underline{v}_i, \bar{v}_i] = [\underline{v}_i, \underline{v}_i]$ and we need only solve VMIN1 to find $\mathcal{J}_i \equiv \{\underline{v}_i\}$.

In general, an upper bound $\bar{v}_i$ could be found by solving VMIN0 with the "min" replaced with "max" and the ceiling function replaced by a floor function. This model, called VMAX0, is a difficult non-convex non-linear program, but it is known that an optimal solution to VMAX0 occurs at an extreme point (Bazaraa *et al.* 1993, pp. 107). Hence, we can find all feasible extreme points of VMAX0 by enumerating all basic solutions, and evaluating the objective for all basic feasible solutions. The largest value yields the desired bound. (For the same reason of computation speed advantage in the continuous model, we do not solve VMIN1 with "min" replaced by "max.")

In particular, for fixed i and W', we try to solve all 2×2 systems of equations of the form

$$\begin{aligned} \boldsymbol{\mu}^T \mathbf{x} &= \breve{\mu}_i \\ \mathbf{w}^T \mathbf{x} &= W'. \end{aligned}$$

If such a 2×2 system has an unique solution, i.e., its determinant is non-zero, we record the solution. If this solution $\hat{\mathbf{x}}$ is feasible, i.e., non-negative, we evaluate the total variance $\mathbf{v}^T \hat{\mathbf{x}}^2$. At the end of the enumeration, the largest total variance found, rounded down, is optimal. The algorithm to implement this procedure is:

Algorithm MAXVAR

Input: $\boldsymbol{\mu}, \mathbf{v}, \mathbf{w}, W', K$ and $\check{\mu}_i$.

Output: Upper bound $\bar{v}_i$ on total variance given $\check{\mu}_i$.

{

    $\mathbf{x} \leftarrow 0$;

    $\bar{v}_i \leftarrow -1$;

    For $(k = 1$ to $K)$ {

        For $(k' = k + 1$ to $K)$ {

            $det = w_k \mu_{k'} - \mu_k w_{k'}$;

            If $(det \neq 0)${

                $x_k = (W' \mu_{k'} - \check{\mu}_i w_{k'})/det$;

                $x_{k'} = (W' - w_k x_k)/w_{k'}$;

                If $((x_k \geq 0)$ AND $(x_{k'} \geq 0))${

                    $\bar{v}_i = \max \{\bar{v}_i, v_k^2 + v_{k'}^2\}$;

                }

            }

        }

    }

    $\bar{v}_i = \lfloor \bar{v}_i \rfloor$;

}

The converse to the case in which $\check{\mu}_i \geq c$, has mean wealth in the smallest range, i.e., $\check{\mu}_i \leq W^2$. In this situation, the objective function of EMIPI is maximized when $\check{v}_j$ is maximized. Thus, $[\underline{v}_i, \bar{v}_i] = [\bar{v}_i, \bar{v}_i]$ and we can find $\bar{v}_i$ by solving "VMAX1" which is the (unlisted) maximizing version of VMIN1.

c. *Step 3*

Here, we compute the standard normal cumulative probability of $\mathscr{Z}^n|(\check{\mu}_i, \check{v}_j)$ for all i-j combinations not eliminated in the previous steps. This is a simple data preparation step for EMIPI.

d.   Step 4

In this step, we determine a lower bound on the optimal objective function value and eliminate all i-j combinations that yield worse objective values. The lower bound corresponds to a "good" feasible solution to EMIPI found by a heuristic solution procedure. First, we compute the objective values of EMIPI for all i-j combinations not eliminated in the previous steps. Next, for each item type $k > 1$, we enumerate all "wealth-consuming" portfolios consisting only of type-k and riskless items, and compute the respective objective values. The maximum objective value found is a lower bound for EMIPI.

e.   Sequence of Execution

The resulting sets $\mathcal{IJ}$ obtained from steps 1 and 2, together with the cumulative probability values computed in Step 3, are applicable to all sub-problems with same initial available wealth W' regardless of time t. Hence, we first carry out the preprocessing steps 1, 2 and 3 to generate the set $\mathcal{IJ}$ for each value of $W' = W^n$, $n = 1, 2, \ldots, N-1$ before executing DMSKP. For each sub-problem in DMSKP with a given available wealth, the algorithm calls the function EMIP($\cdot$), which performs further reduction of the corresponding input set $\mathcal{IJ}$ with preprocessing Step 4 and resolution of EMIPI with a MIP solver. This method of running DMSKP with EMIPI provides an approximate solution to MSKP0, but an exact solution to the discretized problem MSKP1.

E.   COMPUTATIONAL RESULTS

We test DMSKP on several modest-sized problems here. Problem details are listed in Table II. Test Problem 1 is run for three choices of W^n, specifically, $\underline{W}^n$, $0.5(\underline{W}^n + \underline{W}^{n+1})$ and $\underline{W}^{n+1}$, respectively.

The algorithm DMSKP with EMIPI is implemented using the General Algebraic Modeling System (GAMS) (Brooke *et al.* 1996). Computations are performed

Item Types	Test Problem 1 items 1 through 6 from Table I	Test Problem 2 items 1 through 8 from Table I
W_0	30	30
c	80	80
$\underline{W}^{n+1} - \underline{W}^n \ \forall \ n \neq N$	20	10
$W^n \ \forall \ n \neq N$	$\underline{W}^n$ $0.5(\underline{W}^n + \underline{W}^{n+1})$ $\underline{W}^{n+1}$	$0.5(\underline{W}^n + \underline{W}^{n+1})$

Table II. Details of Test Problems 1 and 2.

on a Dell Dimension XPS D333 Pentium II computer with 196 megabytes of RAM. Computational results for all tests are listed in Table III.

$W^n =$	Test Problem 1			Test Problem 2
	$\underline{W}^n$	$0.5(\underline{W}^n + \underline{W}^{n+1})$	$\underline{W}^{n+1}$	$0.5(\underline{W}^n + \underline{W}^{n+1})$
U_0	0.9551	0.9974	1.0000	0.9999
Preprocessing Time	1.54	1.92	1.72	14.07
DMSKP Time	0.87	9.35	16.28	31.97
Total Time	2.41	11.27	18.00	46.04

Table III. Computational results for MSKP1 using DMSKP (Times measured in minutes.)

Using Test Problem 1, we compute the lower and upper bounds on the optimal objective value of MSKP0 by using the pessimistic choice $W^n = \underline{W}^n$ and optimistic choice $W^n = \underline{W}^{n+1}$, respectively. The results are given by the U_0 values in Table III. Furthermore, the results demonstrate the non-linear relationship between U_0 and the choice of W^n: U_0 for $W^n = 0.5(\underline{W}^n + \underline{W}^{n+1})$ exceeds the average of U_0 for $W^n = \underline{W}^n$ and $W^n = \underline{W}^{n+1}$.

The solvability of a problem using DMSKP depends very much on problem size. In particular, solution time using DMSKP grows exponentially with the order of magnitude of the unit means and variances of the items, the total return threshold

40

and the number of discretized wealth levels. We compare the results for both test problems using the choice $W^n = 0.5(\underline{W}^n + \underline{W}^{n+1})$ to get some idea of the effect of increased problem size on the computational effort. In this case, increasing the number of item types by about one third causes a four-fold increase in runtime. So, it does appear that computation times can increase significantly with increased number of item types. In fact, the increase in problem complexity as items types are added depends on the unit weight, and mean and variance in the return of any additional item types. In Test Problem 2, it happens that items of type 8 have the maximum unit variance among all item types. Hence, there is a multiplicative increase in the size of set $\mathcal{IJ}$.

For Test Problem 2, there are 112,906 i-j combinations examined by DMSKP. Without preprocessing, this number would be about 900,000, given that $\bar{\mu} = 128$ and $\bar{v} = 7031$. It is therefore clear that the preprocessing to reduce the set $\mathcal{IJ}$ is critical to the computational efficiency of DMSKP. In addition, further reduction in the size of set $\mathcal{IJ}$ is possible if the unit variance of the item types have a highest common factor a where $a \neq 1$. In such situations, we can eliminate all i-j combinations with $\mod(\tilde{v}_j, a) \neq 0$ and achieve a-fold reduction in the size of $\mathcal{IJ}$, roughly.

IV. CONCLUSIONS AND FUTURE WORK

In this thesis, we have developed new methods for solving certain single- and multi-stage stochastic knapsack problems (SKPs). The particular problems solved are a single-stage integer stochastic portfolio-optimization problem (SPOP) with independence of returns among the various item types (example, stocks, bonds and other financial instruments) in the portfolio, and a multi-stage integer SPOP with inter- and intra-stage independence among item types. For both problems, there is complete dependence of returns within each item type. Additionally, the return for each item type is assumed to follow a normal distribution with known mean and variance. Given an available wealth, we wish to determine a portfolio with the best probability of achieving or exceeding a specified return threshold at the end of the planning horizon. For the single-stage SPOP, portfolio revision is not allowed during the planning horizon, whereas for the multi-stage SPOP, periodic portfolio revision is allowed.

An algorithm called DSSKP is developed to solve the single-stage integer SPOP using a dynamic-programming (DP) approach. DSSKP is implemented using the Java programming language. For a problem from the literature with 11 item types, initial wealth of 30 and return threshold of 60, DSSKP obtains an optimal solution in a fraction of a second on a laptop computer. This efficient, exact method is easy to program; therefore, it is highly portable to different computer platforms. This portability is desirable when DSSKP needs to be run on a more powerful machine for a large-scale problem. We also point out that DSSKP can be easily modified to accommodate bounded variables, which would be important if quantity restrictions were placed on one or more item types.

For the multi-stage SPOP, possible portfolio returns are discretized into N possible levels of wealth for each time period; the nth range is $[W^n, W^{n+1})$, $n = 1, 2, \ldots, N-1$ while the Nth range is $[W^N, \infty)$. Then, any return or wealth value

in $[\underline{W}^n, \underline{W}^{n+1})$ is mapped to a representative value W^n chosen from that range. For a middle-of-the-road approach, we use $W^n = 0.5(\underline{W}^n + \underline{W}^{n+1})$. At each time period but the first, we need to solve N sub-problems to determine an optimal portfolio rebalancing decision for each of N levels of wealth. For period 1, we only need to solve one sub-problem. The recursive resolution of these sub-problems at each time period is handled by a DP-like algorithm, DMSKP. The sub-problems themselves are solved with a mixed-integer programming (MIP) model, EMIPI. Using DMSKP with EMIPI to solve the discretized multi-stage SPOP exactly provides an approximate solution to the original multi-stage SPOP. We obtain lower and upper bounds on the optimal objective value of the original problem through the pessimistic choice $W^n = \underline{W}^n$ and optimistic choice $W^n = \underline{W}^{n+1}$, respectively.

In this study, DMSKP with EMIPI is implemented using the General Algebraic Modeling System (GAMS) (Brooke *et al.* 1996). For a problem with 6 item types, initial wealth of 30, wealth interval of 20 (i.e., $\underline{W}^{n+1} - \underline{W}^n = 20$), and final return threshold of 80, DMSKP with EMIPI obtains an optimal solution to the discretized problem in 11 minutes 17 seconds on a desktop computer. For a larger problem with 8 item types, initial wealth of 30, wealth interval of 10 and final return threshold of 80, DMSKP with EMIPI obtains an optimal solution to the discretized problem in about 46 minutes.

In this thesis, we have shown the relevance and efficiency of DP approaches for solving a single- and multi-stage SKPs for portfolio optimization. It would be more realistic to allow returns to have a general covariance matrix, as in the classical Markowitz portfolio-optimization model (Markowitz 1952, 1959). Hence, extensions of this thesis should investigate techniques to handle:

1. Single-stage SKPs with dependencies among item-type returns,

2. Multi-stage SKPs with intra-stage dependencies among item types, and

3. Multi-stage SKPs with inter-stage dependencies.

The basic model for a single-stage SKP with dependencies among item types would look like SKP1(W), except that $\mathbf{v}^T\mathbf{x}^2$ is replaced by $\mathbf{x}^T\mathbf{V}\mathbf{x}$ where $\mathbf{V}$ is the covariance matrix of the returns from the various item types. An integer NLP method could be applied to solve such a problem. This method could either be a specialized algorithm or an explicit model, to be solved using a commercial solver such as DI-COPT (GAMS Development Corporation, 1999), which in turn calls a MIP solver and an NLP solver. For a problem with a general utility function in the objective function, an approximate model would look like EMIPI. Again, the variance components would need to be modified appropriately. A fractional programming method can be applied to such a problem. But, non-convexities can make it difficult to ensure a global optimal solution when the probability of achieving the return threshold is less than 1/2. (See Henig 1990 and Geoffrion 1967.)

Multi-stage SKPs with intra-stage dependencies, but with inter-stage independence among item types might be solved in a manner similar to that of Chapter III in this thesis. That is, the overall problem would be solved with a DP recursion and each sub-problem would be solved by a method developed for a single-stage SKP with dependencies among item types.

For multi-stage SKPs with inter-stage dependencies, a general model is difficult to solve for it must handle all kinds of uncertainties, including fluctuation of interest rates, changes in the economic health of various business sectors, etc. A limited type of inter-stage dependence in the multi-stage SPOP could be handled by redefining the state variable: A state would represent a certain range of wealth together with a "state of the economy," or "economy" for short. For each economy, we would have a different set of portfolio options available: Industrial stocks might have high mean returns in one economy and low returns in another, but agricultural stocks would look the same in both scenarios. A Markovian model could govern the transitions from economy e at time t to economy e' at time $t+1$ with known probability $p_{ee't}$. DMSKP and EMIPI can be easily modified to handle such models, at least in theory.

One time-dependent parameter that could be handled with ease is the cost (weight) of each item type. In this thesis, we assume time-independent deterministic costs for all item types for simplicity in notation. These weights can easily be changed to time-and-state-dependent parameters. Hence, economy in stage t could affect the costs of relevant item types in stage $t + 1$.

Unequal interval widths for the wealth ranges might improve the accuracy of the discretization. One possibility is to define the interval widths for different wealth ranges such that the probability of achieving wealth in the nth level is the same for all n. Such an approach would have to be approximated, however, since those probabilities depend on the optimal values of the decision variables.

A possible improvement to the discretization approach for the multi-stage SPOP is to view the multi-stage problem as a lattice model and adopt a sampling approach. Specifically, importance sampling could be used to reduce the set of feasible paths examined in the lattice model to a subset which satisfies certain measures of "importance" (Nielsen 1996).

LIST OF REFERENCES

Bazaraa, M. S., H. D. Sherali, and C. M. Shetty, *Nonlinear Programming: Theory and Algorithms*, 2nd ed., John Wiley & Sons, Inc., New York, 1993.

Birge, J. R. and F. Louveaux, *Introduction to Stochastic Programming*, Springer-Verlag, New York, 1997.

Brooke, A., D. Kendrick and A. Meeraus, *GAMS Release 2.25 - A User's Guide*, 2nd ed., GAMS Development Corporation, Washington, D.C., 1996.

Cariño, D. R., T. Kent, D. H. Myers, C Stacy, M. Sylvanus, A.L. Turner, K. Watanabe, and W. T. Ziemba, "Russell-Yasuda Kasai Model: An asset/Liability Model for a Japanese Insurance Company using Multistage Stochastic Programming," *Interfaces*, **24**, pp. 29-49, 1994.

Carraway, R. L., T. L. Morin, and H. Moskowitz, "Generalized Dynamic Programming for Stochastic Combinatorial Optimization," *Operations Research*, **37**, pp. 819-829, 1989.

Carraway, R. L., R. L. Schmidt, and L. R. Weatherford, "An Algorithm for Maximizing Target Achievement in the Stochastic Knapsack Problem with Normal Returns," *Naval Research Logistics*, **40**, pp. 161-173, 1993.

Dreyfus, S. E., and A. M. Law, *The Art and Theory of Dynamic Programming*, Academic Press, Inc., New York, 1977.

GAMS Development Corporation, "DICOPT," [http://www.gams.com/solvers/dicopt/main.htm], March 1999.

Geoffrion, A. M., "Solving Bicriterion Mathematical Programs," *Operations Research*, **15**, pp. 39-54, 1967.

Henig, M. I., "Risk Criteria in the Stochastic Knapsack Problem," *Operations Research*, **38**, pp. 820-825, 1990.

Hiller, R. S. and J. Eckstein, "Stochastic Dedication: Designing Fixed Income Portfolios Using Massively Parallel Benders Decomposition," *Management Science*, **39**, pp. 1422-1438, 1993.

Kall, P. and S. W. Wallace, *Stochastic Programming*, John Wiley & Sons, Inc., Chichester, 1994.

Markowitz, H. M., "Portfolio Selection," *Journal of Finance*, **7**, pp. 77-91, 1952.

47

Markowitz, H. M., *Portfolio Selection: Efficient Diversification of Investments*, John Wiley & Sons, Inc., New York, 1959.

Mitten, L. G., "Composition Principles for Synthesis of Optimal Multistage Processes," *Operations Research*, **12**, pp. 610-619, 1964.

Morton, D. P., and R. K. Wood, "On a Stochastic Knapsack Problem and Generalizations," in D.L. Woodruff, editor, *Advances in Computational and Stochastic Optimization, Logic Programming and Heuristic Search: Interfaces in Computer Science and Operations Research*, pp. 149-168, Kluwer Academic Publishers, 1998.

Nielsen, S. S., "Importance Sampling in Lattice Pricing Models," paper presented at the INFORMS Computer Science Technical Section Conference, 5th, Dallas, Texas, 8 January 1996.

Pyle, D. H. and S. J. Turnovsky, "Safety-First and Expected Utility Maximization in Mean-standard Deviation Portfolio Analysis," in Vickson, R. G. and W. T. Ziemba, editors, *Stochastic Optimization Models in Finance*, pp. 235-241, Academic Press, 1975. (reprinted from The Review of Economics and Statistics, Vol. LII No. 1, February 1970)

Sharpe, W. F., *Portfolio Theory and Capital Markets*, McGraw-Hill Book Company, New York, 1970.

Steinberg, E., and M. S. Parks, "A Preference Order Dynamic Program for a knapsack Problem with Stochastic Rewards," *Journal of the Operational Research Society*, **30**, pp. 141-147, 1979.

Taha, Hamdy A., *Operations Research: An Introduction*, 5th ed., Macmillian Publishing Company, New York, 1992.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center 2
   8725 John J. Kingman Road., Ste 0944
   Ft. Belvoir, VA 22060-6218

2. Dudley Knox Library 2
   Naval Postgraduate School
   411 Dyer Rd.
   Monterey, CA 93943-5101

3. Professor R. Kevin Wood 6
   Department of Operations Research
   Naval Postgraduate School, Code OR/Wd
   Monterey, CA 93943-5008

4. Professor David P. Morton 1
   Department of Mechanical Engineering
   University of Texas At Austin
   Austin, TX 78712

5. Head Librarian 1
   1 Depot Road
   #02-01
   Defence Technology Tower A
   DTT Library
   Singapore 109679
   Republic of Singapore

6. Mr. Khoo, Wai Gea 3
   Operations Research (Department)
   Advance Development (Division)
   DSO National Laboratories
   20 Science Park Drive
   Singapore 118230
   Republic of Singapore